CSE 258 Assignment 1
Purchase Prediction and Rating Prediction
By: Joni DeGuzman (PID: A53212113)

1 Introduction

This assignment involved building recommender systems to solve a purchase prediction task and a rating prediction task on an Amazon dataset that included user reviews of Clothing, Shoes, and Jewelry purchases.

For the purchase prediction recommender system, we had to predict whether a user purchased an item given a (user,item) pair from the test data. This is a binary prediction classification task where a purchased item is given a label 1 and a non-purchased item is given a label of 0.

For the rating prediction recommender system, we had to predict the rating a user gave to a purchased item given a (user,item) pair from the test data. This is a regression task to predict the actual user's rating on an item. The predicted rating needed to be as close as possible to the actual rating which is done by minimizing the mean squared error.

2 Approach

2.1 Purchase Prediction

For the purchase prediction task, I added onto the baseline solutions we implemented in homework 3. I experimented with collaborative filtering techniques to measure item-to-item similarity. Given a target user-item pair from my validation set, I look at the set of all users (U_i) in my training dataset that had purchased this particular item in the past. For each user, I obtain all of their purchases and compute the Jaccard similarity between each purchase and the target item. If similarity is greater than a threshold of 0.12, I predict a purchase. I actually computed a similarity table for all item pairs in the training dataset during my training. This way, during validation, I can simply look up the item-item pair and return the similarity value. I found the average of the similarity values in the similarity table to be 0.1398, so I made the similarity threshold 0.12.

I also computed a similarity table for all user pairs in the training dataset using Jaccard similarity. I found that user-user similarity did not work as well as item-item similarity and decided to just use item-item similarity. I found that using item-to-item similarity measures to determine purchase prediction did not help much in terms of improving my accuracy and could not be used as stand alone model.

My best model combined the popularity model and common category model with item-to-item similarity. For the popularity model, my threshold for predicting an item purchase was if it was in the 40th percentile of popularity. The common category model was what we implemented in the homework except I changed it to predict a purchase if a target item has 2 or more categories in common with any item that a user has purchased in the past.

I used the same training and validation data split from the homework along with my own generated 100000 non-purchased user-item pairs to determine the best model. Once I settled on the best model, I retrained the model to use all 200000 data points. This time, I set my item-to-item similarity threshold as 0.03 because the average similarity value was

PP-2017 V2.1

0.0588. When evaluated on the user-item pair of the test set, I first check if a the target item is above the similarity threshold. If it is not, then I check if it has a greater than 1 common category with the user's previous purchases. If it fails the common category, I check if it is popular. The item is predicted as not being purchased if it fails all of these or if it is a new item.

2.2 Rating Prediction

For the rating prediction task, I created a hybrid model that used the model from the homework that incorporated user and item biases (Model 1) with a matrix factorization model that used singular value decomposition (Model 2),

Model 1: rating(user,item) = $\mu + \beta_u + \beta_i$

Model 2: rating(user,item) = SVD(K, α, λ)

where: μ is the mean, β_u are the user biases, β_i are the item biases, K is the number of latent factors, α is the learning rate, and λ is the regularization parameter.

I used the method of alternating least squares to find the α, β_u, β_i that gave me convergence on the MSE of Model 1. My regularization term in this model was λ = 5. Additionally, I clip the predicted ratings to have the values in the range of $[1, 5]$.

Model 2 was a matrix factorization model using SVD. I used the Surprise Python library for this model because it already has a SVD predication model algorithm that performs training and validation. The SVD model has tunable parameters for number of latent factors, learning rate, and regularization as well as other parameters. I tuned only the above three parameters and chose the following values for my best model:

$$K = 20$$
$$\alpha = 0.008$$
$$\lambda = 0.1$$

I trained each model separately using the first half of the data and evaluated on the second half of the data. Once I settled on my parameters and hyper parameters, I retrained the models on the entire dataset.

In order to compute the predicted rating of a user-item pair on the hybrid model, I used a weighted average of the predictions I obtain from each model. Since SVD is prone to overfitting, I gave more weight to the predicted rating from Model 1.

$$rating = w_1 \times prediction_1 + w_2 \times prediction_2$$
$$w_1 = 0.75$$
$$w_2 = 0.25$$

I also experimented with the rating prediction by putting both models' predicted values into a linear regressor to get the theta parameters to determine final predicted value. Here the feature matrix is:

$$X_i = [1, prediction_1, prediction_2]$$

I found that using the linear regressor to make the predicted rating gave a worse prediction than using a weighted average of the predictions. I got an MSE of around 1.18 while the MSE using the weighted average was about 1.14.

My best model was the hybrid model that utilized the weighted average predicted rating. I still performed worse in the rating prediction task and dropped about 200 spots on the private leaderboard. My hybrid model overfitted to the training data. In hindsight, I should have chosen a better regularizer to prevent overfitting.

3 Ideas for Improvement

If I were to improve my purchase prediction model to obtain a better accuracy, I would implement the one-class recommendation model to maximize the difference between purchased and non-purchased items. I would also construct a feature matrix with the following features:

$$X_i = [1, a, b, c]$$

where:

a = is item popular

b = avg Jaccard similarity of target item to user's previous purchases

c = average number of common categories target item has with user's previous purchase

I would put this feature matrix into an SVM classifier in order to get the optimal prediction decision boundary of purchases and non-purchases. In my approach, I manually tuned the thresholds for best model.

If I were to improve my rating prediction model to obtain a better performance, I would carefully choose my regularizer to prevent overfitting. Looking back, the two models used in the hybrid model was not that different since the matrix factorization already incorporated the model that used just the user and item biases. I could get rid of one of the models and just do a better job of tuning the parameters and hyperparameters of just one model.

I talked to a student that mentioned having different lambda values for the user and item biases. I would use that student's approach to find two different lambda values.